Towards a Deeper Understanding of Nonmonotonic Reasoning with Degrees

Marjon Blondeel *
Advisors:
Steven Schockaert †, Dirk Vermeir ‡, Martine De Cock §

1 Introduction

Answer set programming (ASP) [Baral, 2003] is a form of declarative programming that can be used to model combinatorial search problems in a concise and declarative manner. One of the advantages of ASP is the fact that it works nonmonotonically. On the contrary, classical logic works monotonically: when new knowledge is added, the set of conclusions that can be inferred grows. Hence it is not really suitable to imitate human reasoning since humans constantly revise their knowledge when new information becomes available. In ASP, nonmonotonicity is obtained by using a special operator "not", the *negation-as-failure* operator. An expression of the form $\mathrm{not}\, a$ is "true" when we fail to derive a, whereas $\neg a$, the negation of a, is "true" if we can derive $\neg a$.

For example, consider the following ASP program.

r_1 : suitable $\leftarrow$ certificate $\wedge$ not criminal-record
r_2 : certificate $\leftarrow$ true

Rule r_1 informally means that a person is suitable for a job if there is no reason to think that he has a criminal record and if we can establish that he has a certificate. A rule such as r_2 is called a fact; the head "certificate" is unconditionally true. Given such a program, the idea is to find a minimal set of literals that can be derived from the program. These "answer sets" then correspond to the solutions of the original search problem.

Unfortunately, ASP is not directly suitable for expressing continuous optimization problems since it is limited to expressing problems in Boolean logic. For example, suppose one wants to travel by car from one city to another in winter. The driving time that is needed to do this depends on several factors; for instance the amount of snow, the distance and the traffic. These concepts are a matter of degree rather than Boolean properties, thus we cannot directly use ASP to model this problem. One solution to this problem is to allow propositions to be true to a certain degree in $[0, 1]$ and to generalize the syntax and semantics of ASP using fuzzy logics [Hájek, 1998]. We can then write a rule

$$\text{driving time} \leftarrow f(\text{snow}, \text{distance}, \text{traffic})$$

where "driving time", "snow" , "distance" and "traffic" now have to be seen as atoms that can be assigned a degree in $[0, 1]$. The function f defines how these degrees have to be combined to establish the driving time.

Fuzzy answer set programming (FASP) (see [Blondeel *et al.*, 2013c] for an introduction to the topic) is a generalization of answer set programming (ASP) based on fuzzy logics. In FASP, a continuous search problem is translated into a set of rules $\alpha \leftarrow \beta$ where the body β and the head α are built from literals, expressions of the form $\mathrm{not}\, a$ with a a literal, constants and connectives that can in principal be interpreted by arbitrary $[0, 1]^n \rightarrow [0, 1]$-mappings. Such a rule now intuitively means that the truth degree of α must be greater than or equal to the truth degree of β. Answer sets of FASP programs are then mappings from the set of literals into $[0, 1]$.

Since it is a relatively new concept, little is known about the computational complexity of FASP and almost no techniques are available to compute answer sets of FASP programs. Furthermore, the connections of FASP to other paradigms of nonmonotonic reasoning with continuous values are largely unexplored. In our dissertation, we contribute to the ongoing research on FASP on two different levels:

- **Complexity issues (Section 2)** We have pinned down the complexity of the direct syntactical generalization of classical ASP to FASP, and we have developed an implementation into bilevel linear programming for this type of programs. For another class of FASP programs with more syntactic freedom, we showed a connection to an existing open problem about the complexity of integer equations, indicating that settling the complexity of FASP programs in this case will not be an easy task.
- **Connection to fuzzy modal logics (Section 3)** We combined the paradigms of fuzzy logic and autoepistemic logic into fuzzy autoepistemic logic, and showed that the latter generalizes FASP. Furthermore, we have introduced generalizations of classical propositional modal logics of belief, and we have obtained a generalization of a known result on the relationship between stable expansions, belief sets and "only believing" operators. Our

*Vrije Universiteit Brussel, Department of Computer Science, Pleinlaan 2, 1050 Brussel, Belgium, mblondee@vub.ac.be, Funded by a joint Research Foundation-Flanders (FWO) project
†Cardiff University, School of Computer Science and Informatics, 5 The Parade, Cardiff, CF24 3AA, s.schockaert@cs.cardiff.ac.uk
‡Vrije Universiteit Brussel, Department of Computer Science, Pleinlaan 2, 1050 Brussel, Belgium, dvermeir@vub.ac.be
§Ghent University, Department of Applied Mathematics, Computer Science and Statistics (S9), Krijgslaan 281, 9000 Gent, Belgium, martine.decock@ugent.be

ongoing work deals with fully developing this "only believing" logic to obtain a better understanding of the nature of fuzzy autoepistemic logic and hence of FASP.

2 Complexity issues

We have analyzed the computational complexity of FASP under Łukasiewicz semantics in [Blondeel *et al.*, 2013b]. Łukasiewicz logic is a particular kind of fuzzy logic that is often used in applications because it preserves many desirable properties from classical logic. Given a FASP program P, a literal l and a value $\lambda_l \in [0,1] \cap \mathbb{Q}$, we are interested in the following decision problems:

1. **Existence**: Does there exist an answer set I of P?
2. **Set-membership**: Does there exist an answer set I of P such that $I(l) \geq \lambda_l$?
3. **Set-entailment**: Is $I(l) \geq \lambda_l$ for each answer set I of P?

For programs with rules of the form

$$a_1 \oplus \ldots \oplus a_n \leftarrow b_1 \otimes \ldots \otimes b_m \otimes \text{not } c_1 \otimes \ldots \otimes \text{not } c_k$$

where $\oplus$ and $\otimes$ are resp. the Łukasiewicz disjunction and conjunction, we showed that the complexity of the main reasoning tasks is located at the first level of the polynomial hierarchy although for classical ASP it is located at the second level. This is due to the fact that we can use linear programming to verify yes instances of the decision problems in polynomial time. Moreover, we provided a reduction from reasoning with such FASP programs to bilevel linear programming, thus opening the door to practical applications.

For the subclass of programs in which negation-as-failure is not allowed and where there is at most one literal in the head of each rule, we showed that the answer set can be found in polynomial time. Surprisingly, when allowing disjunctions to occur in the body of rules – a syntactic generalization which does not affect the expressivity of ASP in the classical case – the picture changes drastically. Reasoning tasks are then located at the second level of the polynomial hierarchy and for simple FASP programs, i.e. programs in which negation-as-failure and negation do not occur and in which each rule has exactly one atom in the head, we have not been able to pin down the complexity yet. More specifically, we have an algorithm to find the unique answer set of these programs in pseudo polynomial time but whether this problem is NP-complete remains an open problem. Moreover, the connection to an existing open problem about the complexity of integer equations [Bjorklund *et al.*, 2003] suggests that the problem of fully characterizing the complexity of FASP in this more general setting is not likely to have an easy solution. Another possibly even more intriguing and fundamental open research question is whether there exists a FASP program in which each head contains at most one literal without answer sets. Other future work is to study the complexity under other semantics.

3 Connection to fuzzy modal logics

Autoepistemic logic is an important formalism for nonmonotonic reasoning. It extends propositional logic by offering the ability to reason about an agent's (lack of) beliefs. More precisely, these beliefs are a set of sentences in a propositional language augmented by a modal operator B. Moreover, autoepistemic logic is well known to generalize the stable model semantics of ASP [Gelfond and Lifschitz, 1988]. In [Blondeel *et al.*, 2013a] we combined the ideas of autoepistemic logic and fuzzy logic to a fuzzy autoepistemic logic which can be used to reason about one's beliefs in the degrees to which properties are satisfied. We showed that, like in the classical case, fuzzy autoepistemic logic generalizes FASP.

On the other hand, there are well known links between autoepistemic logic and several nonmonotonic modal logic systems. In ongoing work we are investigating how fuzzy autoepistemic logic can be studied in many-valued modal settings. In particular, we have introduced generalizations of classical propositional modal logics of belief based on finitely-valued Łukasiewicz logic for which we obtained completeness w.r.t appropriate Kripke style semantics and for which we could prove NP-completeness for the satisfiability problem. We obtained a generalization of Levesque's [Levesque, 1990] result on the relationship between stable expansions, belief sets and "only believing" operators. In future work, by fully developing this "only believing" logic we want to obtain a better understanding of the nature of fuzzy autoepistemic logic and hence of FASP.

References

[Baral, 2003] C. Baral. *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press, 2003.

[Bjorklund *et al.*, 2003] H. Bjorklund, S. Sandberg, and S. Vorobyov. Complexity of model checking by iterative improvement: the pseudo-boolean framework. In *Proc. of the 5th Andrei Ershov Memorial Conference "Perspectives of System Informatics"*, pages 381–394, 2003.

[Blondeel *et al.*, 2013a] M. Blondeel, S. Schockaert, M. De Cock, and D. Vermeir. Fuzzy autoepistemic logic and its relation to fuzzy answer set programming. *To appear in Fuzzy Sets and Systems*, 2013.

[Blondeel *et al.*, 2013b] M. Blondeel, S. Schockaert, D. Vermeir, and M. De Cock. Complexity of fuzzy answer set programming under Łukasiewicz semantics. *Submitted*, 2013.

[Blondeel *et al.*, 2013c] M. Blondeel, S. Schockaert, D. Vermeir, and M. De Cock. Fuzzy answer set programming: An introduction. In *Soft Computing: State of the Art Theory and Novel Applications*, volume 291 of *Studies in Fuzziness and Soft Computing*. Springer, 2013.

[Gelfond and Lifschitz, 1988] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In *Proc. of the 5th International Conference and Symposium on Logic Programming*, pages 1070–1080, 1988.

[Hájek, 1998] P. Hájek. *Metamathematics of Fuzzy Logic*. Trends in Logic, 1998.

[Levesque, 1990] H.J. Levesque. All I know: A study in autoepistemic logic. *Artificial Intelligence*, pages 263–309, 1990.